Extreme Heterogeneity Workflow Management

Ewa Deelman, Ph.D.

USC Viterbi
School of Engineering
Information Sciences Institute

https://deelman.isi.edu

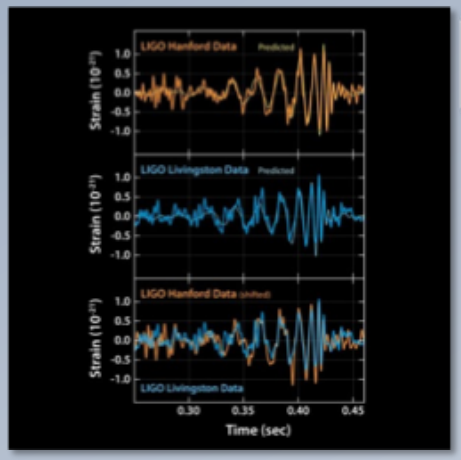
http://pegasus.isi.edu

Advanced LIGO PyCBC Workflow

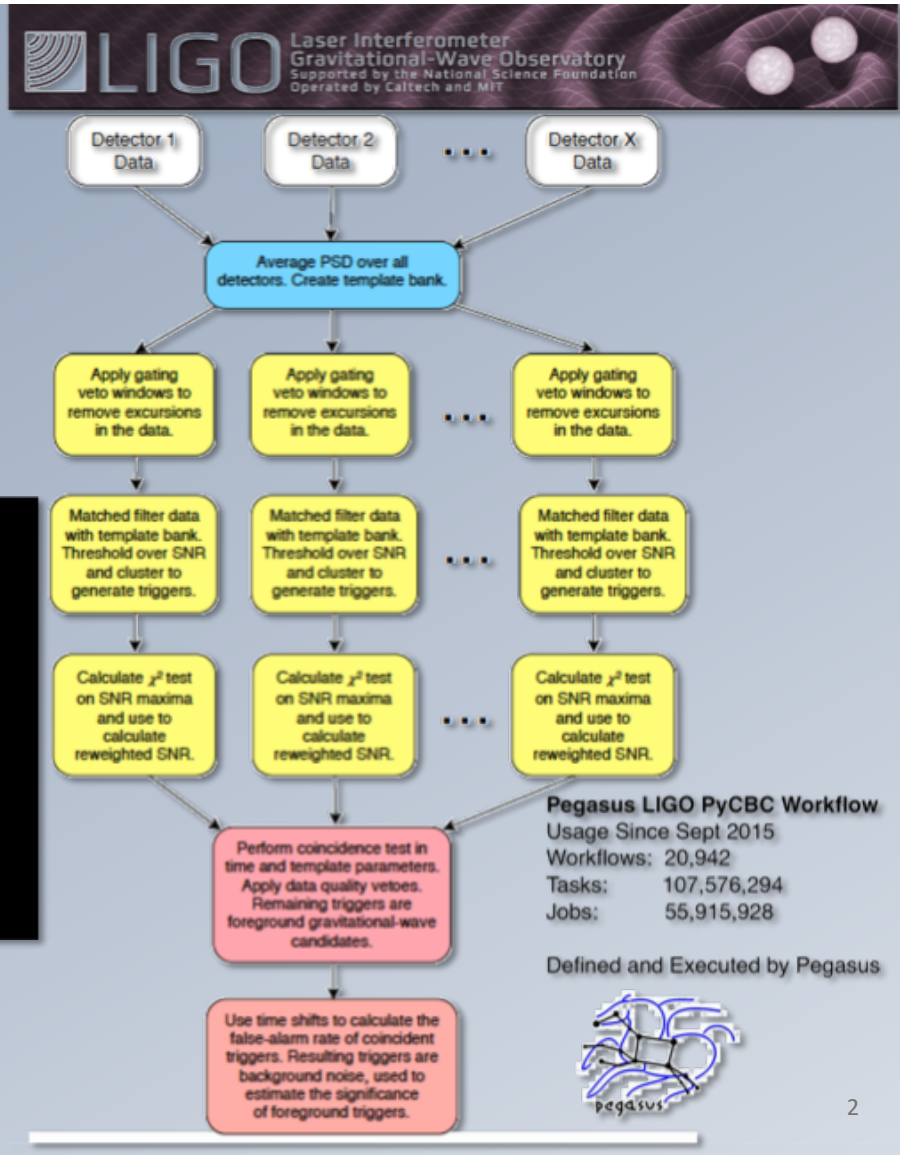
One of the main pipelines to measure the statistical significance of data needed for gravitational wave detection

Contains 100's of thousands of jobs and accesses on order of terabytes of data

Uses data from multiple detectors



LIGO's resources: 11 large clusters at various institutions and affiliates
– Data is replicated at sites in the US and Europe
– Each cluster has Grid middleware and HTCondor installed
– GridFTP used for data transfers



Use our Pegasus software to automate the execution of tasks and data access

Pegasus

http://pegasus.isi.edu

Our Approach: Submit locally, Compute globally, workflow is abstract, help support productivity

Work Definition

Pegasus Workflow Management System

Community Resource

Local HTCondor Pool

LIGO Data Archive

data

work

LIGO HTCondor Pool

LIGO HTCondor Pool

LIGO Data Archive

LIGO Data Archive

LIGO workflows are composed of High Throughput Computing (HTC) jobs

Pegasus

http://pegasus.isi.edu

Easily add new resources because of the workflow abstraction

Work Definition

Pegasus Workflow Management System

Community Resource

Local HTCondor Pool

LIGO Data Archive

data

work

Blue Waters

XSEDE

OSG

LIGO HTCondor Pool

LIGO HTCondor Pool

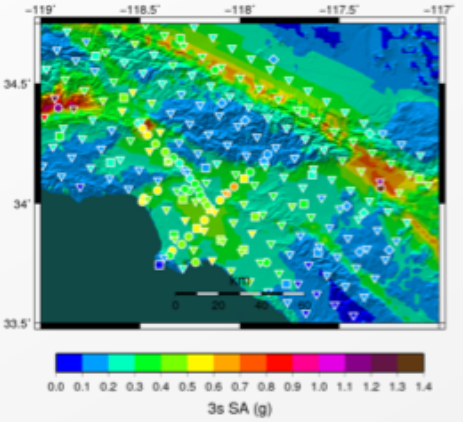
LIGO Data Archive

LIGO Data Archive

Pegasus

Southern California Earthquake Center

Mix of HPC and HTC codes

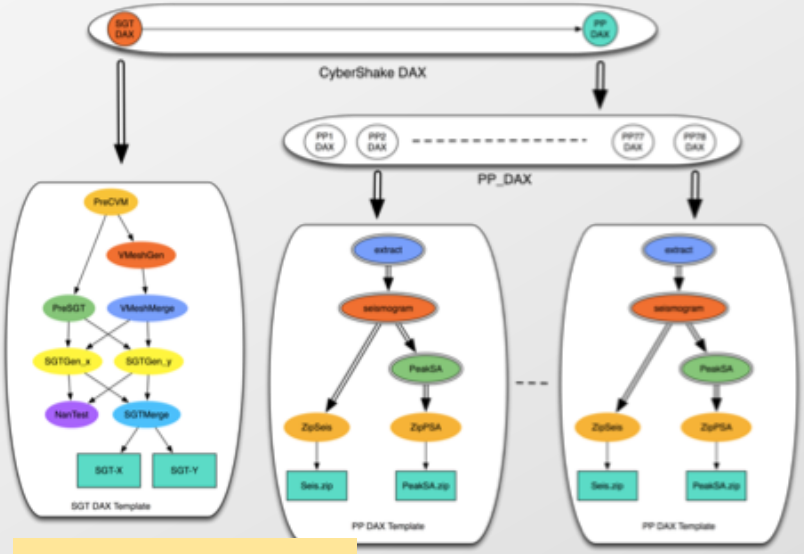


Workflow Ensembles: 239 Workflows

- Each site in the input map corresponds to one workflow
- Each workflow has:
  - ✧ **820,000 tasks**

CyberShake PSHA Workflow

- ❖ **Description**
  - ✧ Builders ask seismologists: “What will the peak ground motion be at my new building in the next 50 years?”
  - ✧ Seismologists answer this question using Probabilistic Seismic Hazard Analysis (PSHA)



Runs on HPC (Titan)

Runs on HTC or HPC

Pegasus

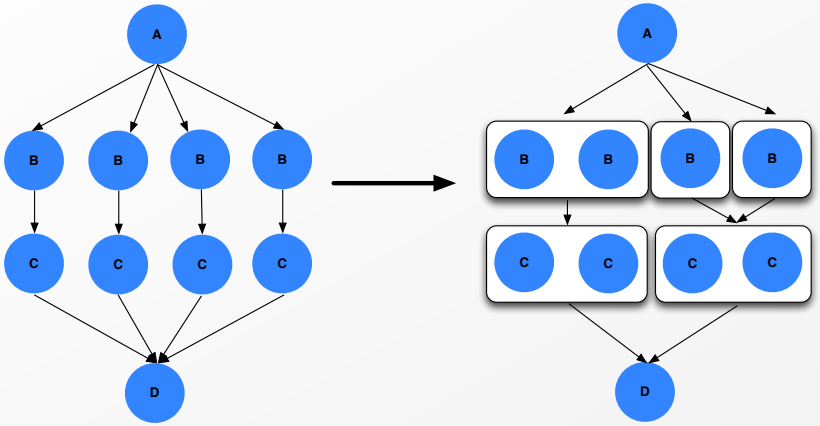
http://pegasus.isi.edu

Workflow Management System (WMS) Functions

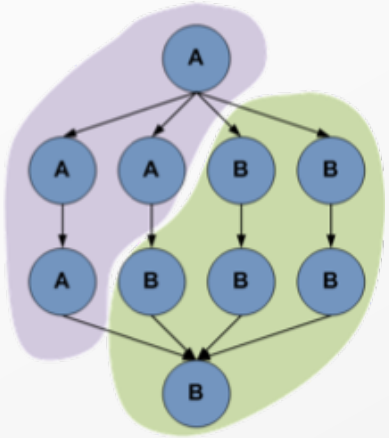
- Automate computational scientific methods that scientists rely on for their work
- Allow scientists to describe their workflows at a scientific level, in a resource-independent way
- Discover what resources (computation, data, software) are available
- Devise a plan and generate an executable workflow:
  - Select the appropriate resources based on an architecture, availability of software, performance, reliability, availability of cycles, storage,..
  - How to best adapt the workflow to the resources
  - What protocols to use to access the data, to schedule jobs
  - What data to save along the way
- Execute the plan
  - In a reliable and scalable way
  - Keep track of what data was accessed, generated and how. (provenance, reproducibility)
- Sometimes WMS provisions resources ahead of the execution

Solutions to running HTC on HPC:

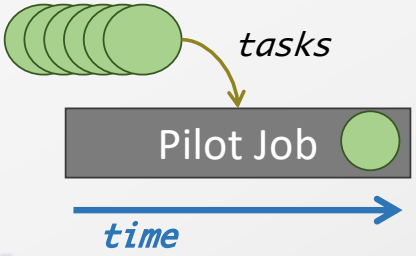
Task clustering



Partition the workflow into sub-workflows and send them for execution to the target system



Use "pilot" jobs to dynamically provision a number of resources at a time



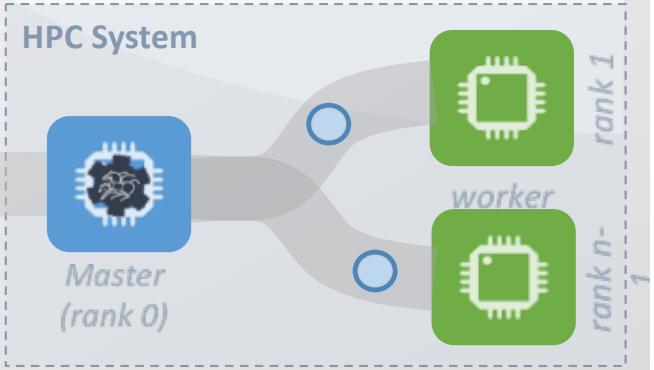
Pegasus

submit host
(e.g., user's laptop)

workflow wrapped as an MPI job

Allows sub-graphs of a Pegasus workflow to be submitted as monolithic jobs to remote resources

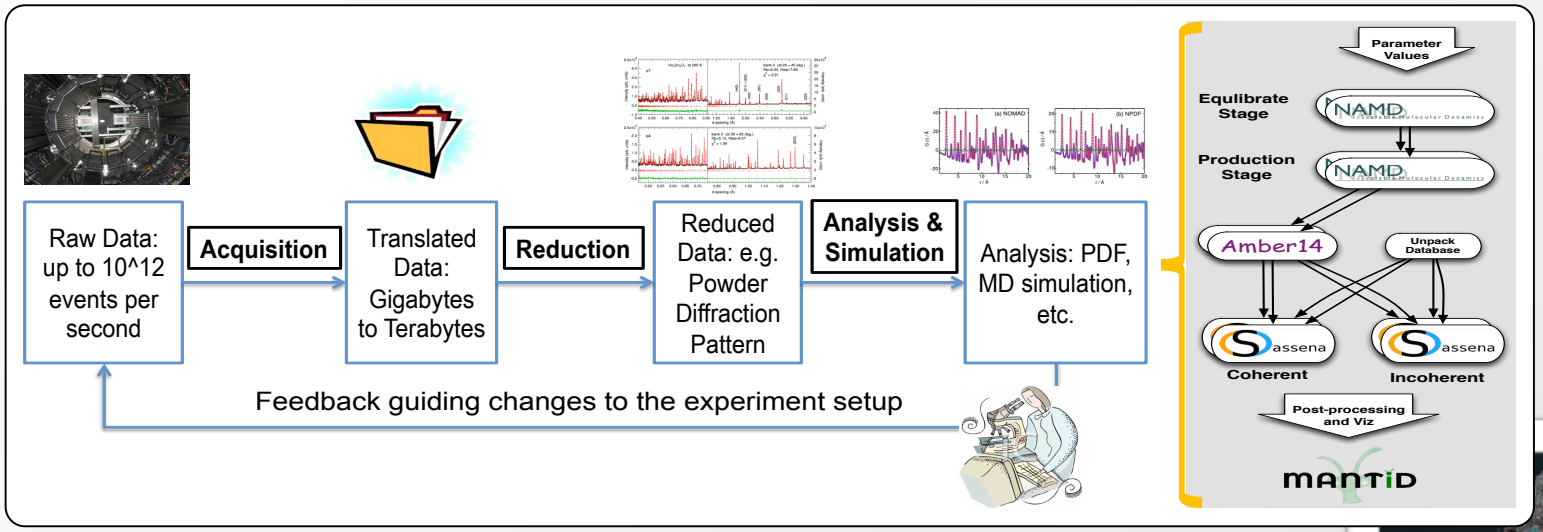
http://pegasus.isi.edu



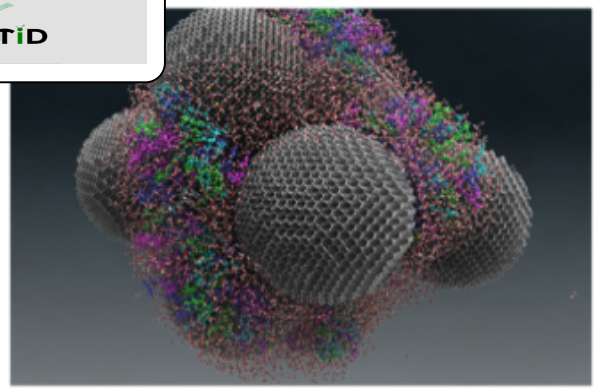
PMC (Pegasus MPI Cluster), a specialized workflow engine

SNS workflow, Instrument in the loop, ORNL

- Time critical, want to impact a physical experiment that is underway



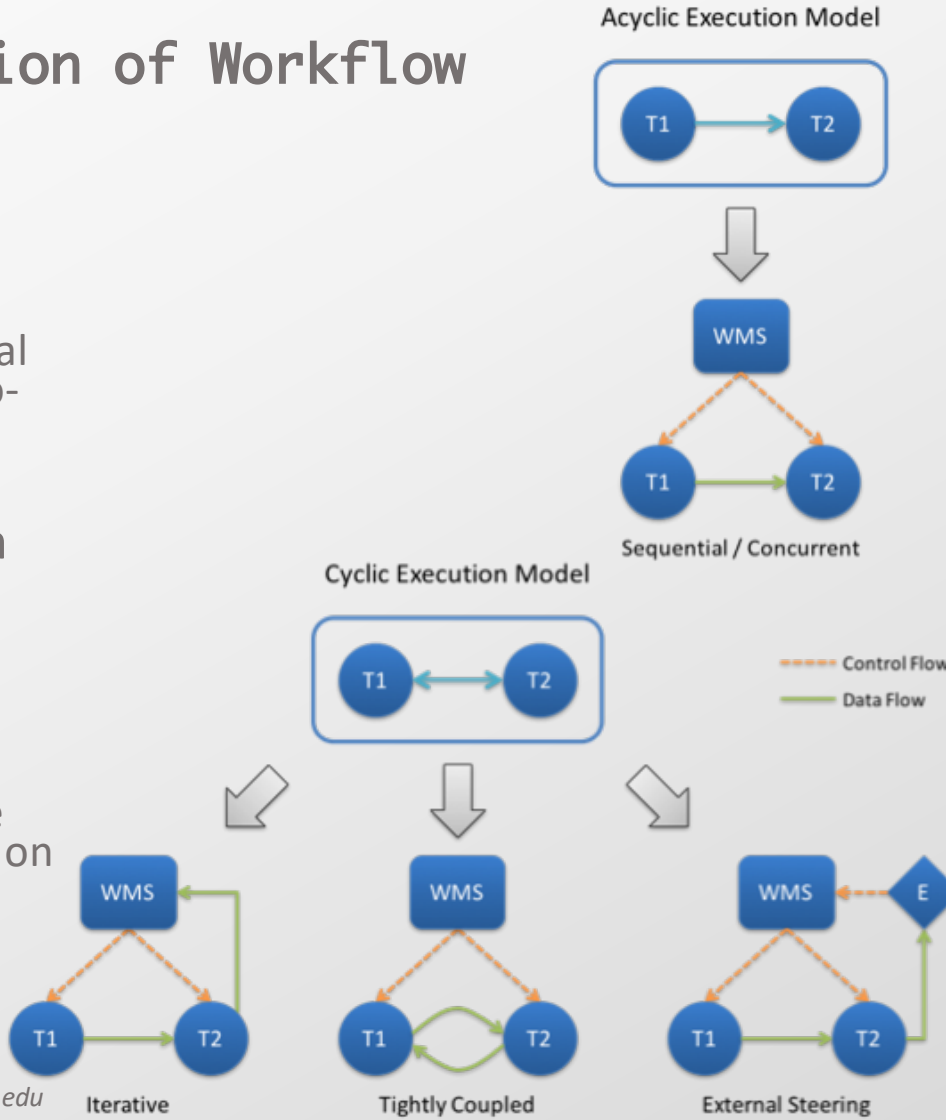
- May want to couple MD &. analytics



Water is seen as small red and white molecules on large nanodiamond spheres. The colored tRNA can be seen on the nanodiamond surface. (Image Credit: Michael Mattheson, OLCF, ORNL)

New Applications Need the Automation of Workflow Management Systems

- Todays workflows
  - Independent high-throughput tasks
  - Loosely couple tasks, where data flow is pipelined
  - Tightly coupled applications that have bi-directional communications and need to be co-located and co-scheduled
  - A mix of all
- Users don't want to worry about where to run
  - need results in a timely manner
  - need ease of design, use, and automation
- Workflow Management Systems
  - + can cross system boundaries
  - + can select the appropriate resources, schedule the needed data movement, send tasks for execution on the target resources
  - + there are systems that couple applications in-situ: Decaf, DataSpaces, ADIOS



Challenges of Workflow Management on Extreme Heterogeneous Systems

- Applications:
  - Tasks can have different resource needs
  - The needs may not be fully known ahead of the execution
  - There may be a number of inter-related workflows running at the same time (workflow ensembles)
- EH Systems:
  - Resource discovery is based on static information about general availability
  - Limited dynamic information from the scheduler
  - Limited resource provisioning capabilities, but there are some good examples
    - Can reserve burst buffers as part of the workflow
    - Can share burst buffers across workflows to support data reuse
- Main issue: who controls the resources?
  - Traditional schedulers do fine-grained resource assignments, may not be scalable or desirable for extreme, highly heterogeneous systems
  - In an alternative view WMS could request a set of resources from the scheduler and manage them on behalf of the workflow or workflow ensemble
    - Doing it indirectly today with pilot jobs

Resource control

- WMS knows the current and potentially future workload needs at a coarse level so it can potentially make better decisions
- Resource discovery for WMS scheduling:
  - Number of available cores, CPUs, GPUs, memory
  - Available of various architectural features, such as burst buffers and their capacity, interfaces
  - Discovery of data location (if you pre-staged data into a bb or node-local disk for example)
- Resource provisioning:
  - Not just focused on compute resources
  - Could be asking for:
    - Compute resources “close to data sets”
    - The same compute resources as job *j-1* had
- Questions?
  - What information to expose about the EH System?
  - How does this information need to be aggregated/what’s the right level of abstraction?

Job scheduling

- Need task performance models that take into account the EH system features
  - Mix of analytical models and simulation
- Need models that take into account data movement within the storage hierarchy: memory, burst buffers, file system, via LAN, via WAN
- Focus on data-aware scheduling for a job, workflow, workflow ensemble
- Need more functionality from schedulers to support changing application demands over time

Online Application and System Monitoring

- Besides information needed for making resource provisioning and scheduling decisions, need information during execution to:
  - Capture performance and faults
  - Potentially discover new data sets
  - Any other information that can help WMS make decisions about releasing/acquiring resources
    - New resources available?
- WMS needs to be self-aware, assure its own execution does not affect the workflows it managed
  - Assuming the WMS or a specialized workflow engine is running in-situ along side the simulation and analytics
- Questions?
  - (WMS) Today's information systems– e.g. Darshan collect a lot of information, how can it be synthesized or analyzed to provide near real-time, actionable information?
  - (WMS) How can ML techniques help?
  - (EH) Is compute resource utilization the best metric to use for evaluation?
  - (EH) How do you divide the responsibility for the health of the application and system between the scheduler and WMS?

Fault Tolerance

- Performance models can be used for anomaly detection
- Need algorithms to quickly detect anomalies
- WMS could adapt, suspend, abort workflow, or trigger an alert
- Need to co-manage the execution of applications, anomaly analysis, WMS
- Need more sophistication from WMS to make scheduling and adaptation decisions
- Need better fault tolerance in HPC/WMS environments
  - Investigate how data replication techniques can be used to improve fault tolerance, while minimizing the impact of energy consumption
  - Explore tradeoffs between data re-computation and data retrieval from DRAM/NVM/disk (time to solution and energy consumption)

Other things we learned in Distributed Area WMS We can apply to HPC application management

- Provenance Capture and Reproducibility: WMS capture provenance information about the creation, planning, and execution
  - Up to now, the approach has been to save everything – problem
  - Provenance capture may need to adapt to the behavior of the application (coarse and fine levels of details, compression)
  - May want to automatically re-run parts of the computation and re-produce the results and a more detailed provenance trail on demand
  - Systems need to be made reproducibility aware: provide enough introspection to reason about the validity and reproducibility of the results

Conclusions

- Need to keep the separation between the workflow description and the executable workflow
- Take into account heterogeneous workflow applications and workflow ensembles
- There is no single WMS that easily traverses HTC and HPC boundaries although scientific applications do
  - Need new workflow technologies that more versatile
  - Explore various data communication modalities within workflows, taking full advantage of system architectural features. (memory, burst buffer, file system)
  - Make sure to include data flow from DTNs and other remote storage
  - Develop better resource provisioning and job scheduling capabilities
- There are opportunities to define new schedulers and new scheduler/WMS interactions
  - Need more information and control flow
  - Need for abstraction for information flow
  - Need to combine resource provisioning and job scheduling
- Not mentioned: need to discuss relationship to runtime systems

Ewa Deelman, deelman@isi.edu